РАЗРАБОТКА КАСКАДНЫХ КЛАССИФИКАТОРОВ ХААРА

Чинда Аделе
Студент
Факультет прикладной информатики
Институт инженерных и цифровых технологий
Белгородский государственный университет
Электронная почта: chindahadele2@gmail.com

Научный руководитель: Маматов Евгений Михайлович
доцент
Кафедра инжиниринга и цифровых технологий
Белгородский государственный университет
Электронная почта: mamatov@bsu.edu.ru*;*

Аннотация: Усовершенствованные обученные классификаторы HAAR-Cascade с исключительной точностью обнаружения для идентификации объектов при различных условиях освещения доступны в Интернете.

Основным недостатком применения таких классификаторов к алгоритмам обнаружения является отсутствие знаний о процессе обучения таких классификаторов, невозможность изменить параметры обучения и набор данных, используемый для разработки классификатора.

В статье приводится систематический процесс разработки каскадных классификаторов ХААРА с использованием программы Cascade Trainer. Наборы данных с положительными и отрицательными выборками генерируются для целей обучения и тестирования, а переобучение классификаторов демонстрируется после анализа матрицы ошибок после каждой фазы обнаружения.

Ключевые слова — ХААР-Каскадный классификатор; положительные выборки; отрицательные выборки; Графический интерфейс, Обучающий набор данных; Тестовый набор данных; Ложноположительный; Ложноотрицательный;

I. ВВЕДЕНИЕ

Алгоритм каскадного обнаружения объектов HAAR предназначен для обнаружения объектов на изображениях или видео в реальном времени. Алгоритм использует функции обнаружения краев или линий, разработанные Виолой и Джонсом в их исследовательской статье 2001 года "Быстрое обнаружение объектов с использованием усиленного каскада простых функций". Для дальнейшего обучения алгоритму дается большое количество положительных изображений, состоящих из экземпляров обнаруживаемого

объекта, и большое количество отрицательных изображений без экземпляра обнаруживаемых объектов.

В этой статье обсуждается, как разрабатывать классификаторы с использованием очень простого в использовании графического инструмента под названием "Cascade Trainer GUI" (созданного Амином Ахмади). Графический интерфейс — Cascade Trainer - это приложение для обучения, тестирования и улучшения моделей каскадных классификаторов. Он устанавливает настройки с помощью графического интерфейса и упрощает использование инструментов OpenCV для обучения и тестирования классификаторов.

II. ГРАФИЧЕСКИЙ ИНТЕРФЕЙС КАСКАДНОГО ТРЕНАЖЕРА

Запуск приложения-классификатора приведет к появлению начальной страницы, как показано ниже.

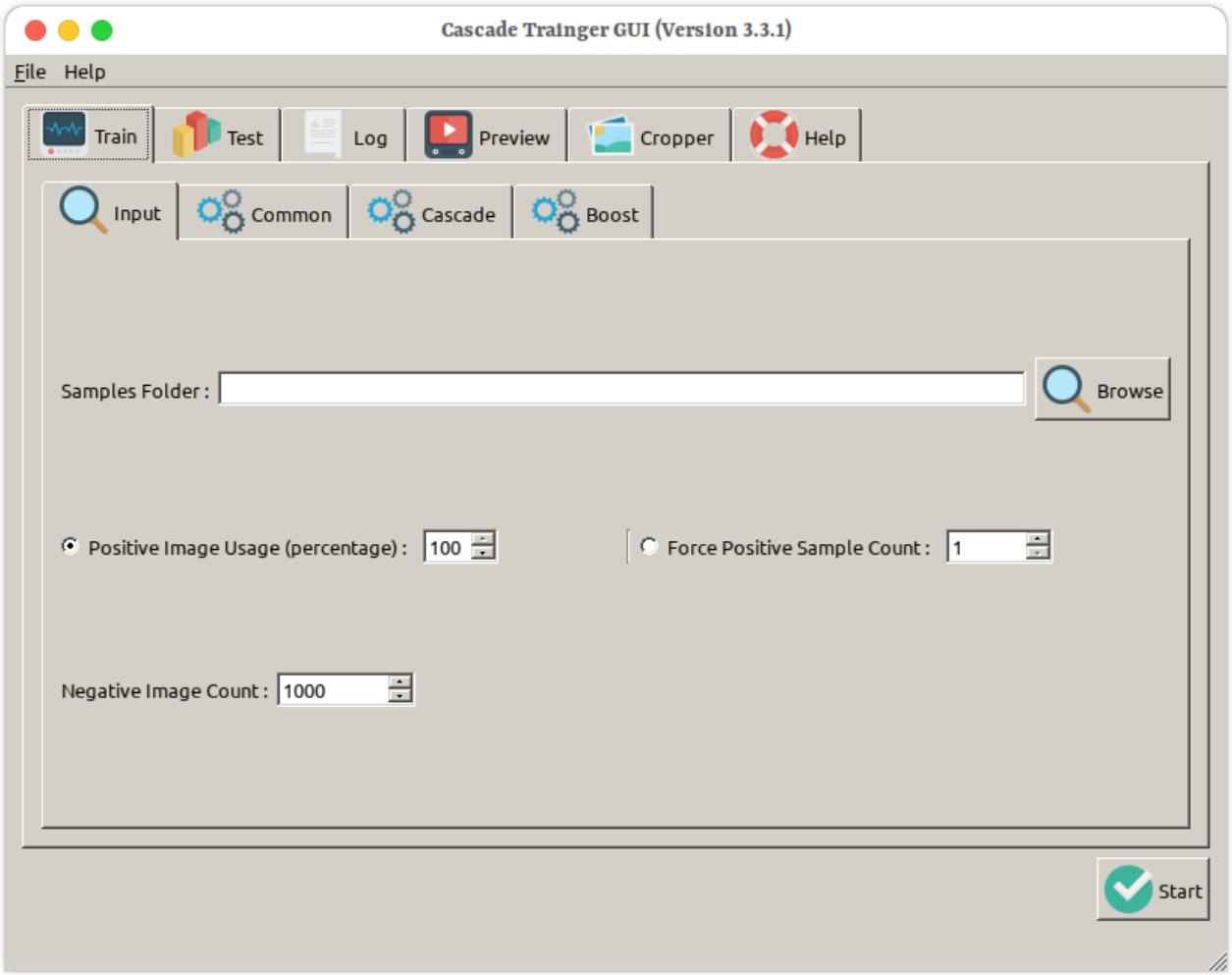

Рисунок 1. Графический интерфейс каскадного тренажера

На Рисунок 1 показаны многочисленные функции приложения. Классификаторы могут быть обучены с использованием первого экрана. Пользователь предоставляет десятки тысяч образцов набора данных, включая

как положительные, так и отрицательные примеры, для обучения классификаторов. Изображения или видео с экземплярами объектов, для которых обучается классификатор, составляют положительные наборы данных. Отрицательные наборы данных включают изображения или видео, в которых нет никаких экземпляров объектов, для которых обучается классификатор.

Чтобы начать обучение, для классификатора должна быть создана папка, а внутри нее созданы две другие папки. Одна из вложенных папок называется "p", поскольку она будет содержать положительный набор данных, а другая - "n" для хранения отрицательного набора данных.

Для этой статьи была создана папка под названием "Классификатор 1", а внутри нее папки для положительных и отрицательных наборов данных. Алгоритм, описанный в статье, обучен распознавать лица с носовыми масками. Таким образом, папка "p" содержит изображения лиц с включенными масками, в то время как папка "n" содержит изображения, кроме изображений лиц с масками. Наборы данных можно получить из Google Dataset или из Kaggle.
Процесс запускается нажатием кнопки Обзор на вкладке Поезд, чтобы выбрать папку для классификатора.

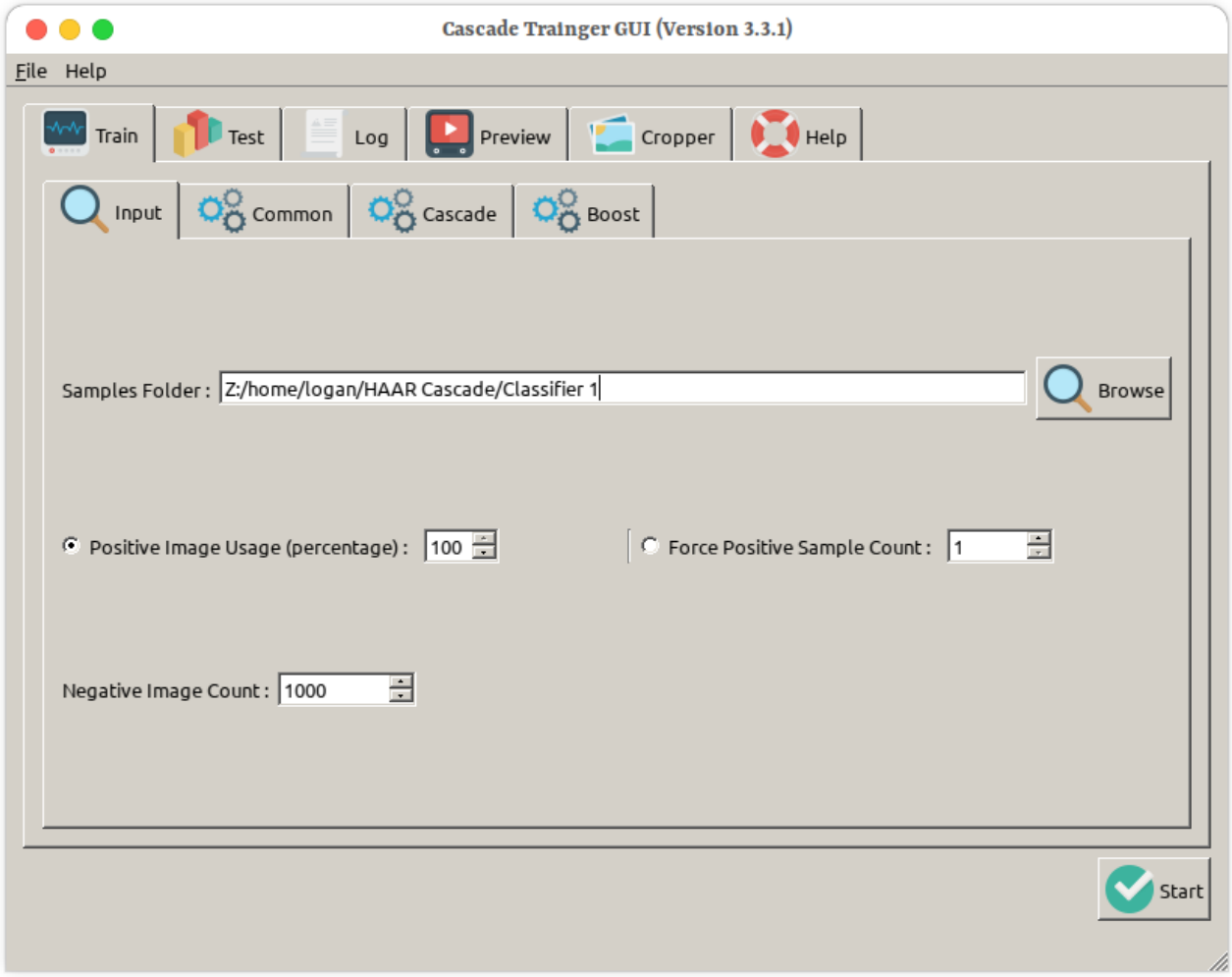

Рисунок 2. Вкладка “Ввод” (на вкладке "Train")

Другие настройки вкладок остаются неизменными, а затем открывается вкладка "общие". Вкладки Common, Cascade и Boost можно использовать для настройки обучения классификатора путем выбора различных параметров.

Графический интерфейс Cascade Trainer по умолчанию устанавливает наиболее оптимальные и рекомендуемые настройки для этих параметров. Однако мы все равно должны изменять некоторые параметры для каждой итерации обучения. В этом исследовании мы не предоставили подробного описания всех этих факторов, что выходит за рамки данного исследования и требует глубокого понимания методов каскадной классификации.

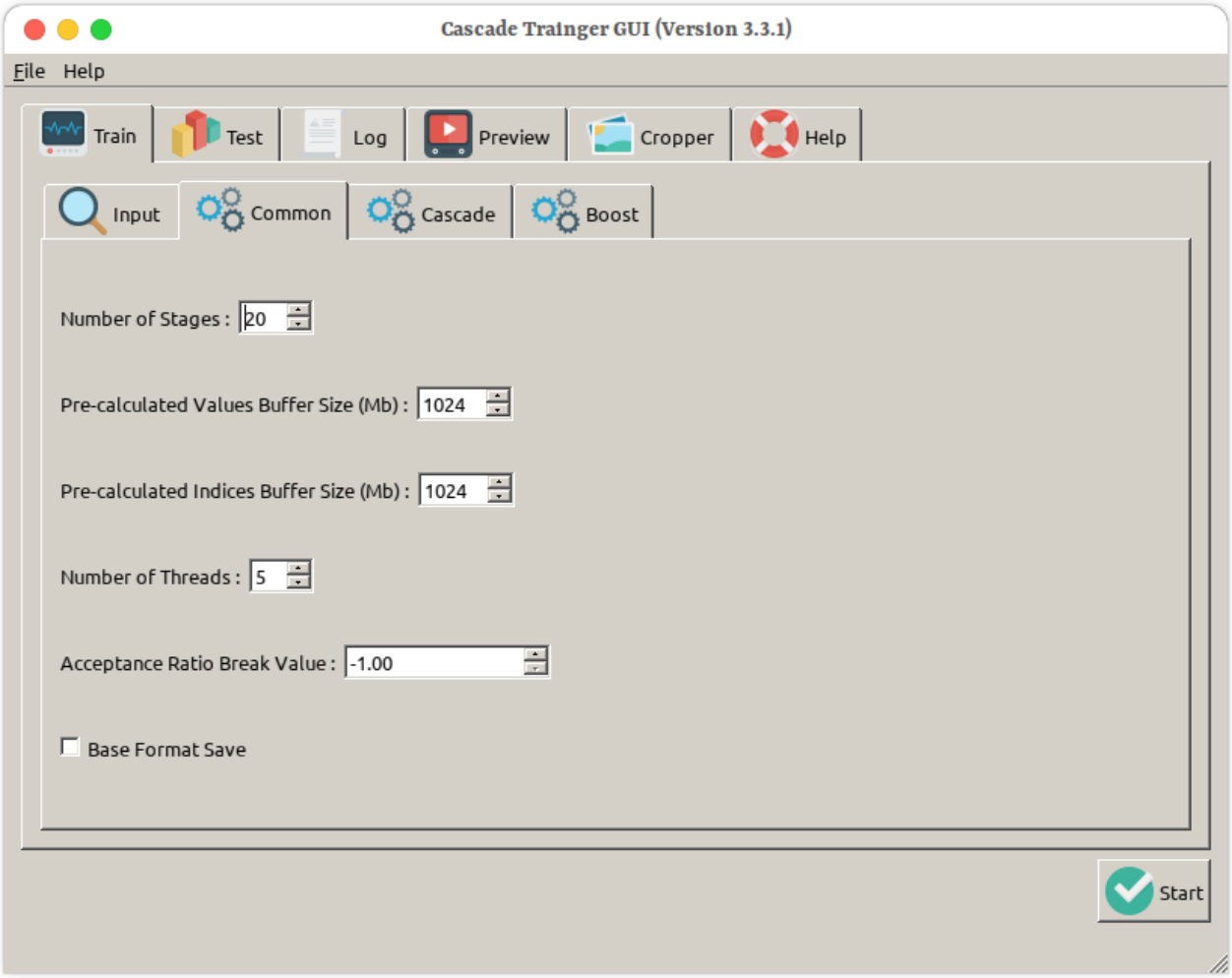

Рисунок 3. Вкладка “Common” (на вкладке "Train")

Здесь определены тренировочные повторы. Как правило, чем больше итераций, тем выше точность обнаружения классификатора. Однако по мере увеличения числа итераций увеличивается и время, необходимое для разработки и обучения классификатора. Затем можно увеличить объем буфера предварительного вычисления, чтобы ускорить процесс обучения. Им может

быть выделено много оперативной памяти, но нужно быть осторожным, чтобы не выделять слишком много или слишком мало. Например, если на компьютере установлено 4 ГБ оперативной памяти, было бы безопасно установить оба размера буфера, показанные ниже, равными 2048. Затем на вкладке "Каскад" указываются ширина и высота выборки. Рекомендуемые настройки для ширины и высоты выборки заключаются в том, чтобы оставить один аспект на 24 и соответствующим образом установить другой.

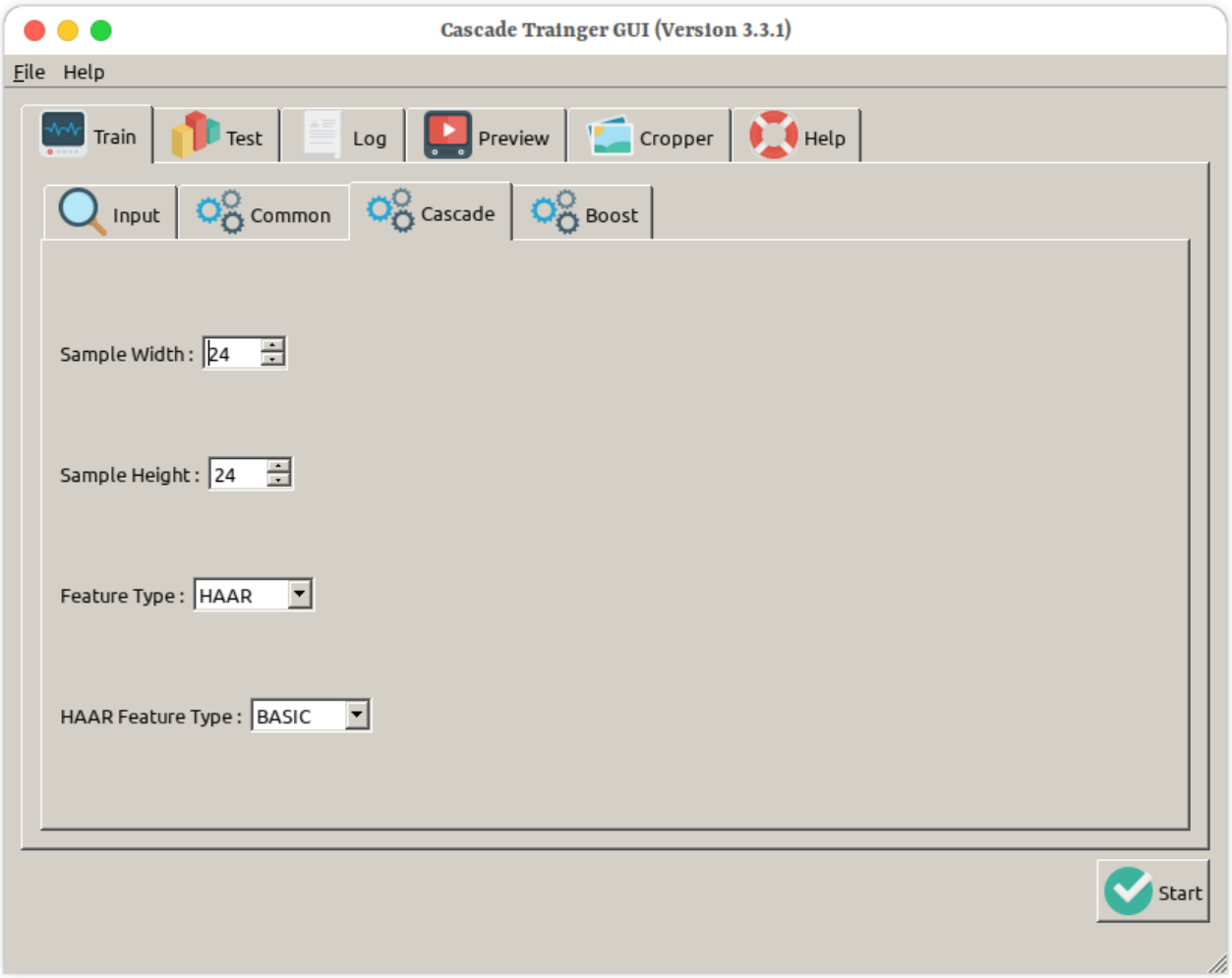

Рисунок 4. Вкладка “Cascade” (на вкладке "Train")

Тип объекта может быть выбран между HAAR и LBP. HOG используется только в том случае, если используется OpenCV 3.1 или более поздняя версия. Классификаторы HAAR довольно точны, но требуют много времени для обучения; следовательно, гораздо лучше использовать LBP, если классификаторам дается большое количество примеров изображений. С другой стороны, классификаторы LBP менее точны, но при этом обучаются и

обнаруживают почти в три раза быстрее. Параметры на вкладке Boost остаются неизменными.

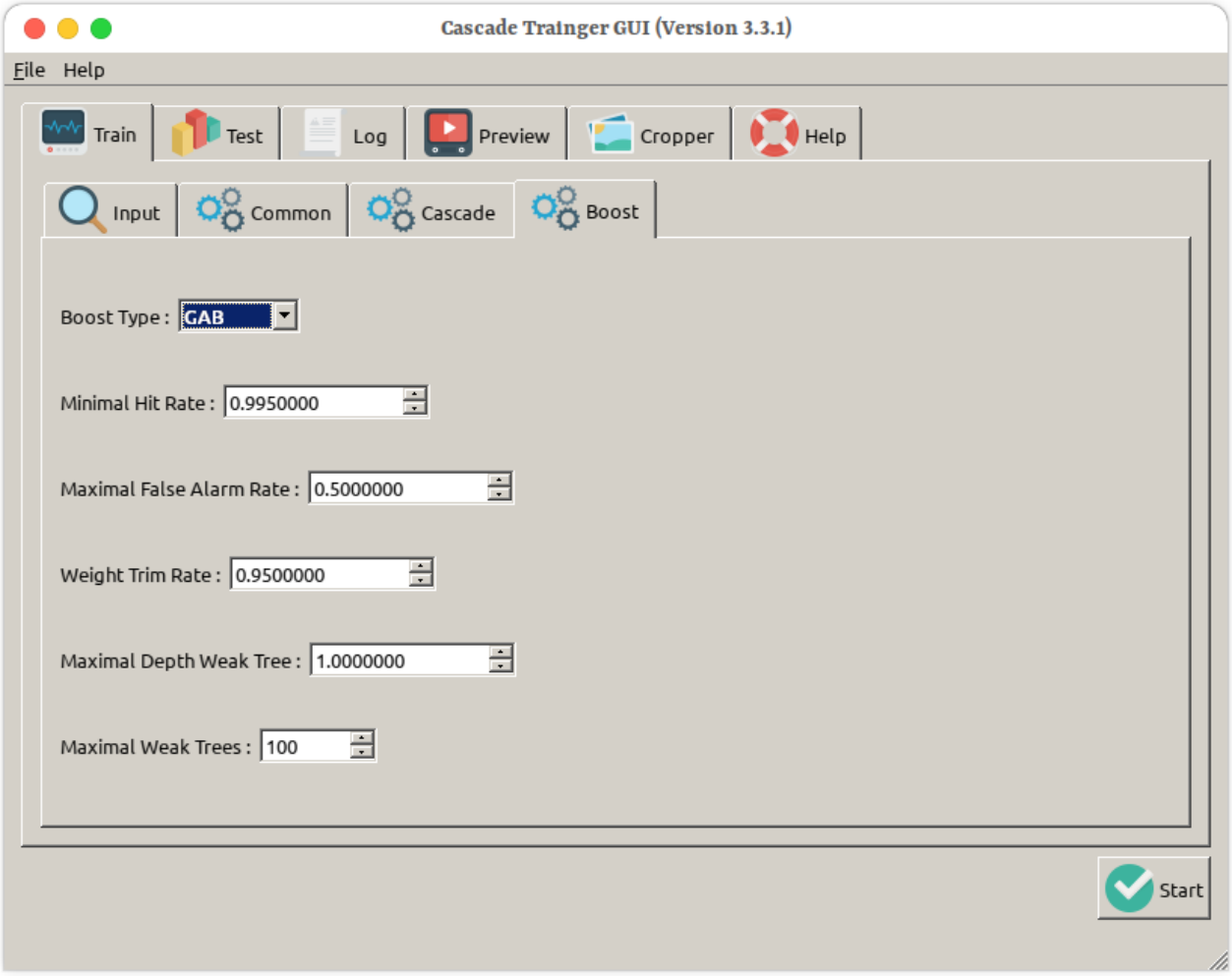

Рисунок 5. Вкладка “Boost” (на вкладке "Train")

III. ПОДГОТОВКА НАБОРА ДАННЫХ

Для каждой операции обнаружения требуется набор положительных и отрицательных фотографий. Каскадный классификатор сообщает "OpenCV", где искать в изображении.

В этой статье используется 60 положительных образцов и 40 отрицательных образцов. Для редактирования наборов данных, полученных из Kaggle, используйте функцию "Обрезки" графического интерфейса Cascade Trainer.

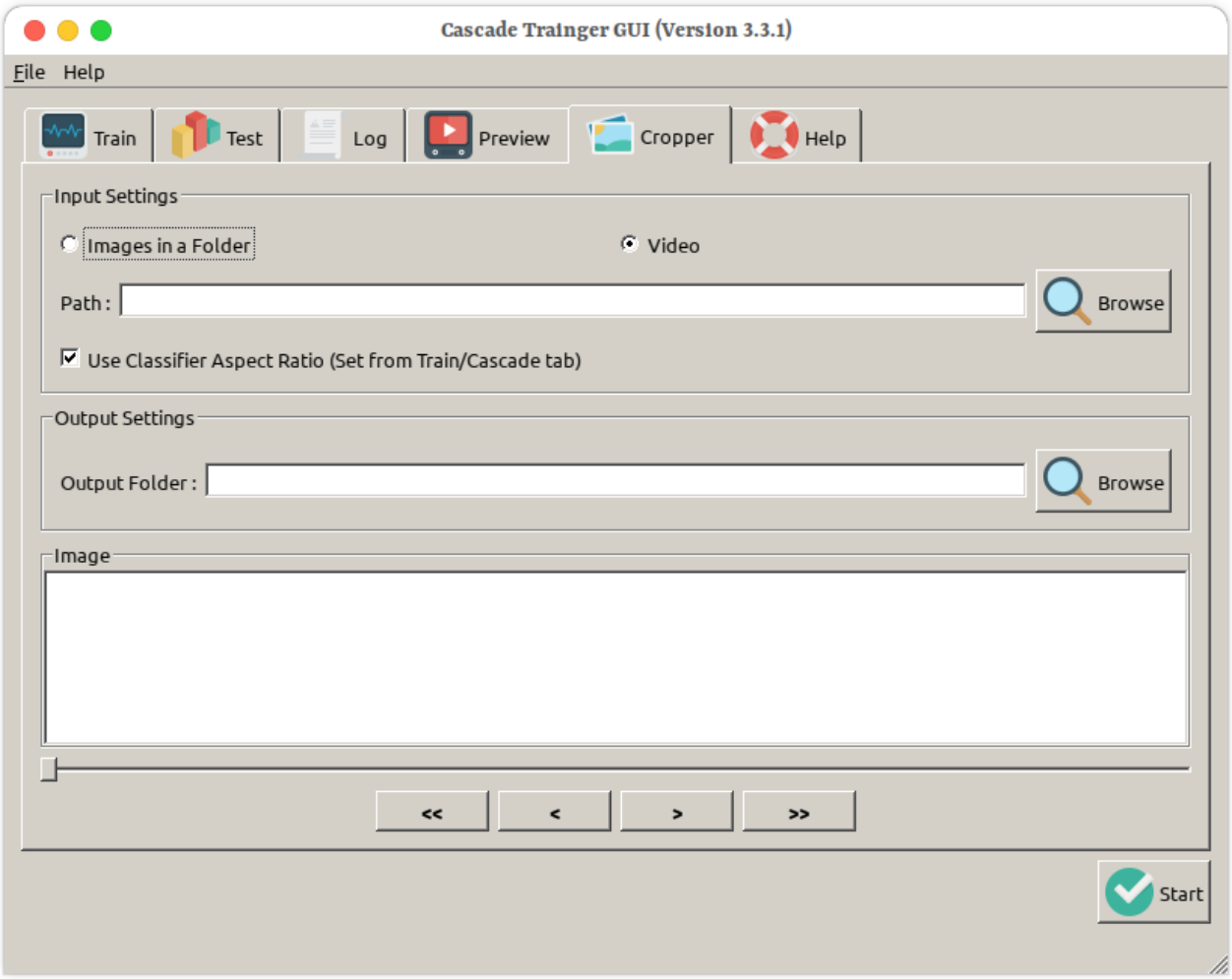

Рисунок 6. Утилита обрезки

Для использования утилиты обрезки выбирается переключатель “изображение в папке”, в поле настройки ввода вводится путь к папке с необработанными положительными тестовыми данными, другие параметры остаются неизменными, а путь к папке “p” указывается в настройках вывода в качестве поля вывода.

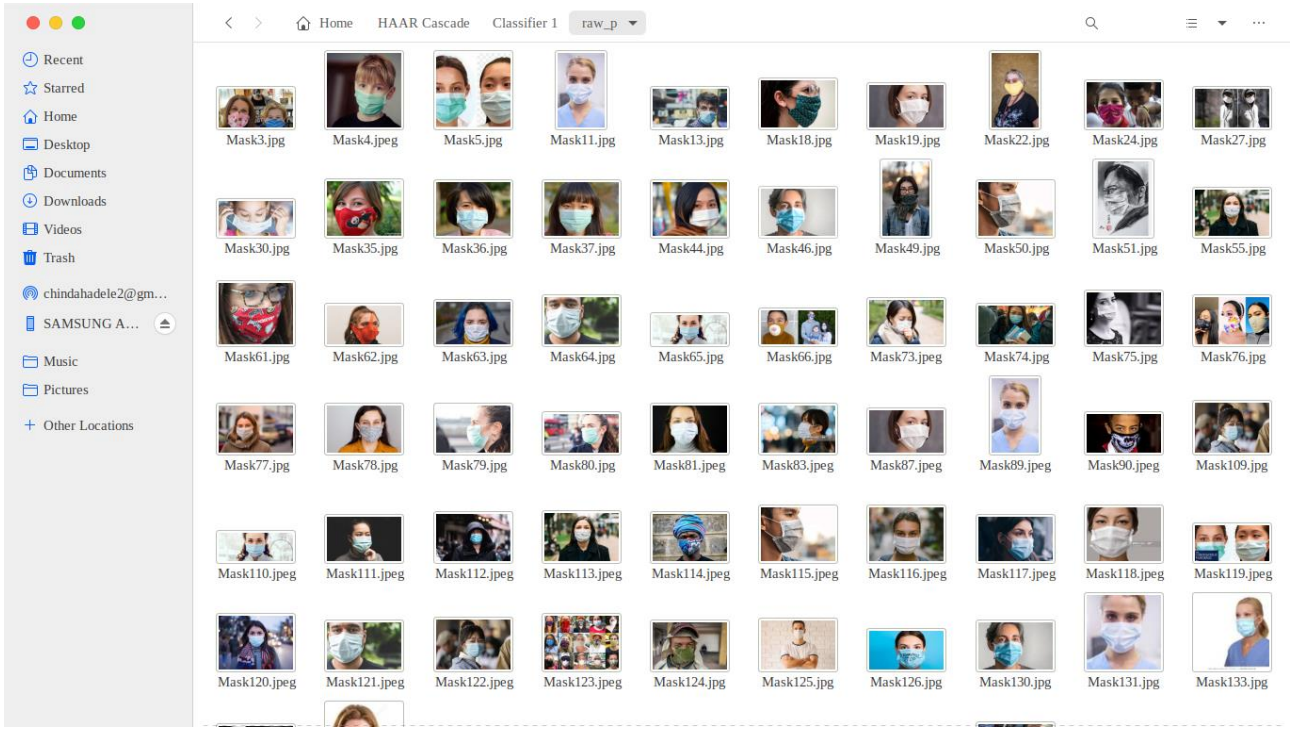

Рисунок 7. Изображения лиц для позитивного набора данных

С помощью кнопок “> <” мы можем выбрать следующее или предыдущее изображение для обрезки. Затем мы указываем путь к папке “p”, которую мы создали в начале. Здесь все обрезанные изображения собираются в ту папку “p”, которая обозначает “Положительные” изображения для каскада. Затем, указав путь, мы начинаем обрезку следующим образом:

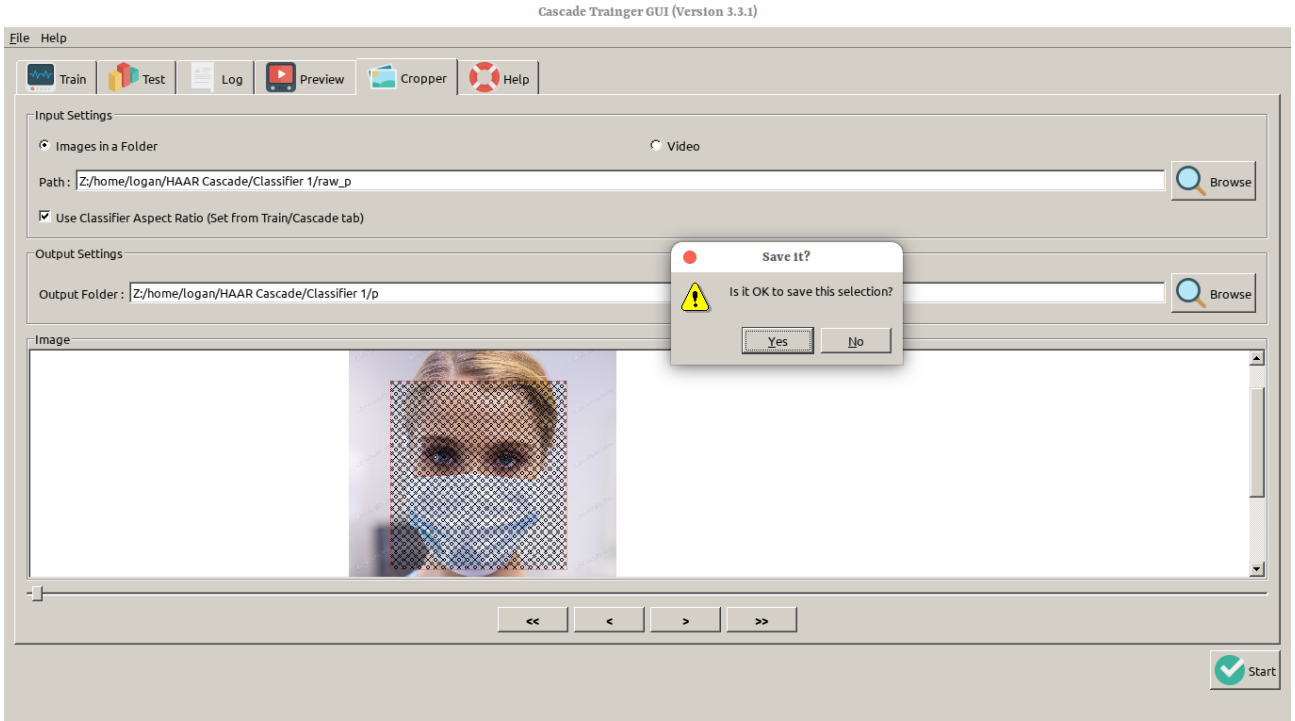

Рисунок 8. Обрезка лицевой части для положительного образца

Здесь только область “лицевая часть” на изображении выделена с помощью мыши, и как только вы остановите выделение, мы получим всплывающее сообщение с надписью “можно ли сохранить это выделение”, и если вы считаете, что это нормально, то сохраните это выделение в папке “p”.
Пожалуйста, обратите внимание: Если вы не чувствуете, что выбранная область не соответствует точному лицу, вы можете отменить это всплывающее окно и повторно выбрать нужную область для сохранения. Вы можете проделывать этот шаг столько раз, сколько сможете, пока не почувствуете, что выделенная область является именно тем объектом, который мы хотим обнаружить. Затем, после создания желаемых положительных выборок (в данном случае 60), мы переходим к созданию набора данных отрицательных выборок, и 40 таких выборок создаются и сохраняются в папке “n”.

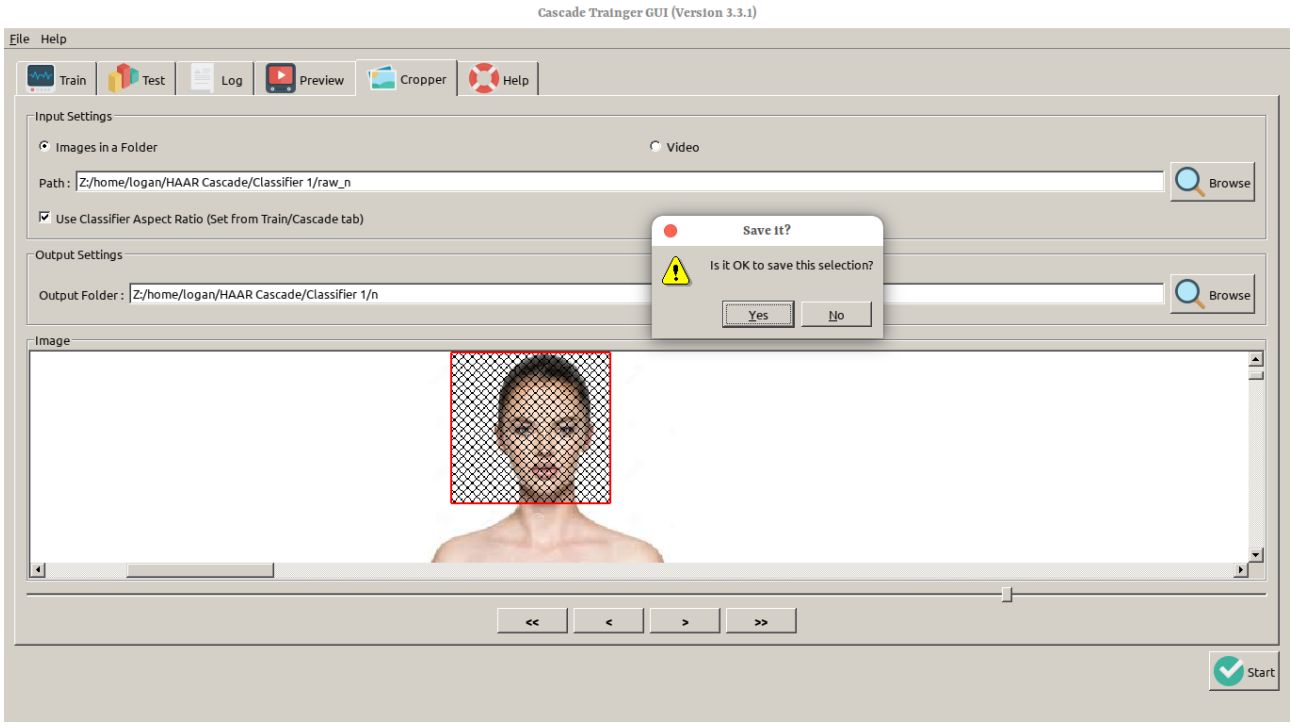

Рисунок 9. Обрезка изображения для негативного образца

IV. СОЗДАНИЕ И ОБУЧЕНИЕ МОДЕЛЕЙ

После установки всех параметров нажмите кнопку "Пуск" внизу, чтобы начать обучение вашего каскадного классификатора. Во время тренировки вы увидите следующий экран журнала.

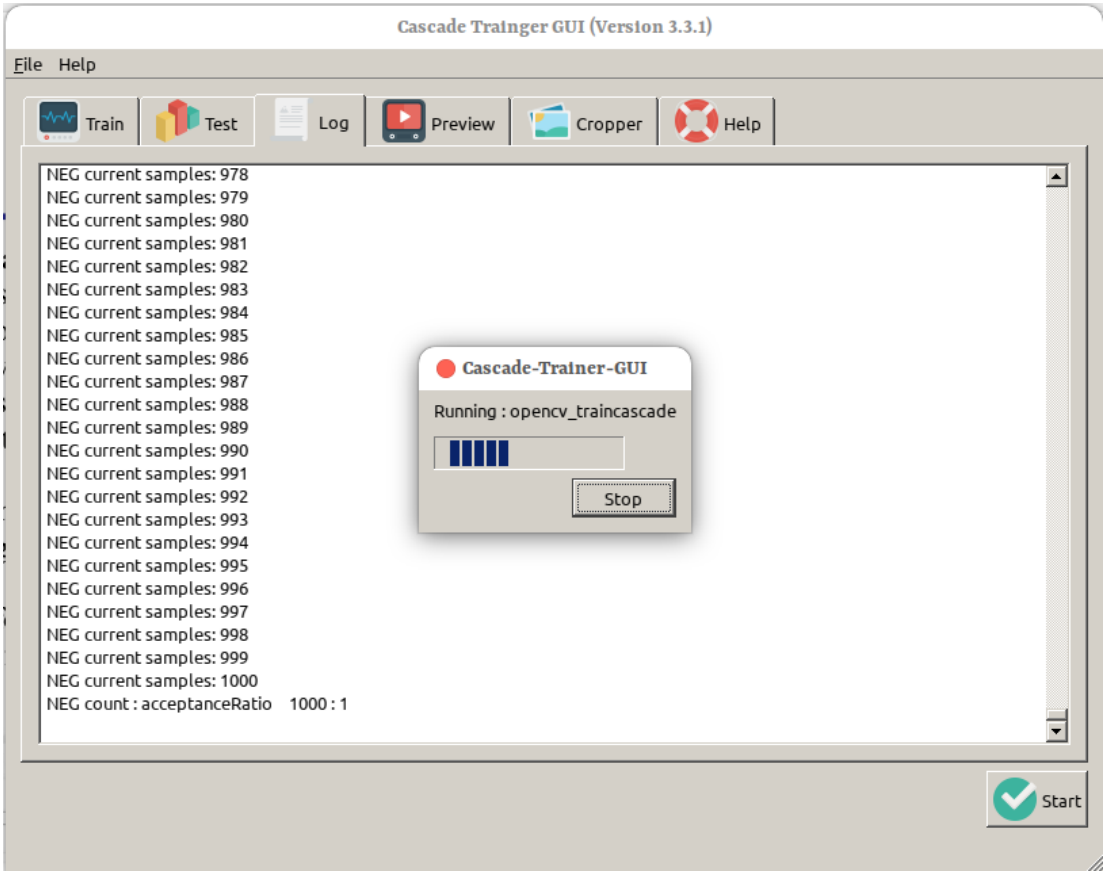

Рисунок 10. Экран журнала (во время обучения)

Дождитесь завершения обучения. Теперь, если мы выйдем из графического интерфейса Cascade Trainer и перейдем в папку classifier, мы заметим, что в этой папке созданы новые файлы и папки.

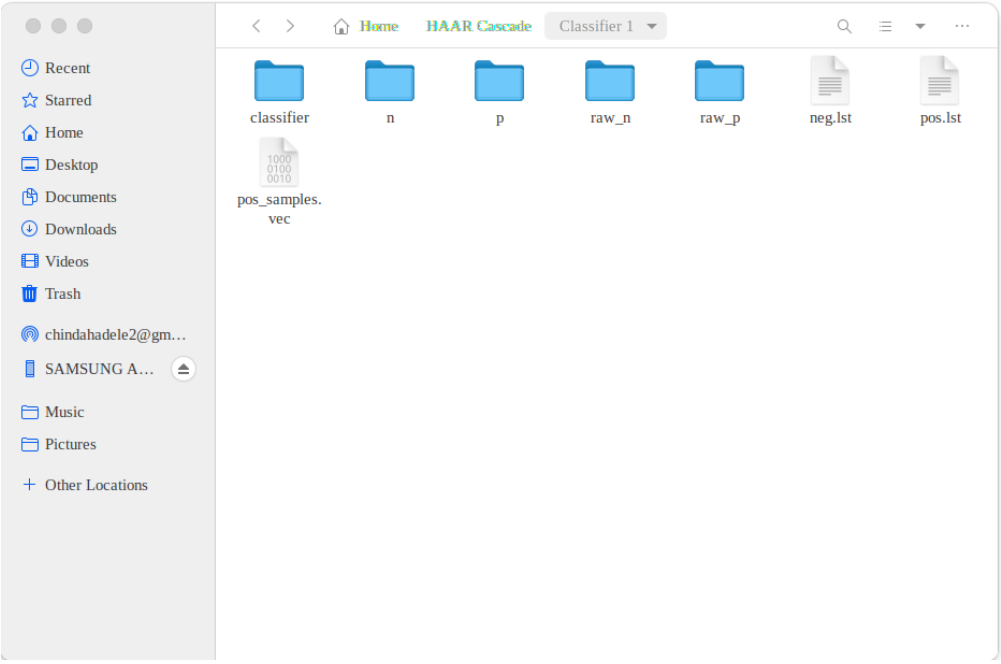

Рисунок 11. Папка пользовательского классификатора (внутри нее созданы новые файлы и папки)

Папки “n” и “p” нам знакомы, но остальные - новые. Папка “Классификатор” содержит XML-файлы, которые создаются на разных этапах обучения. Если мы проверим внутри папки “классификатор”, мы заметим нечто похожее на следующее.

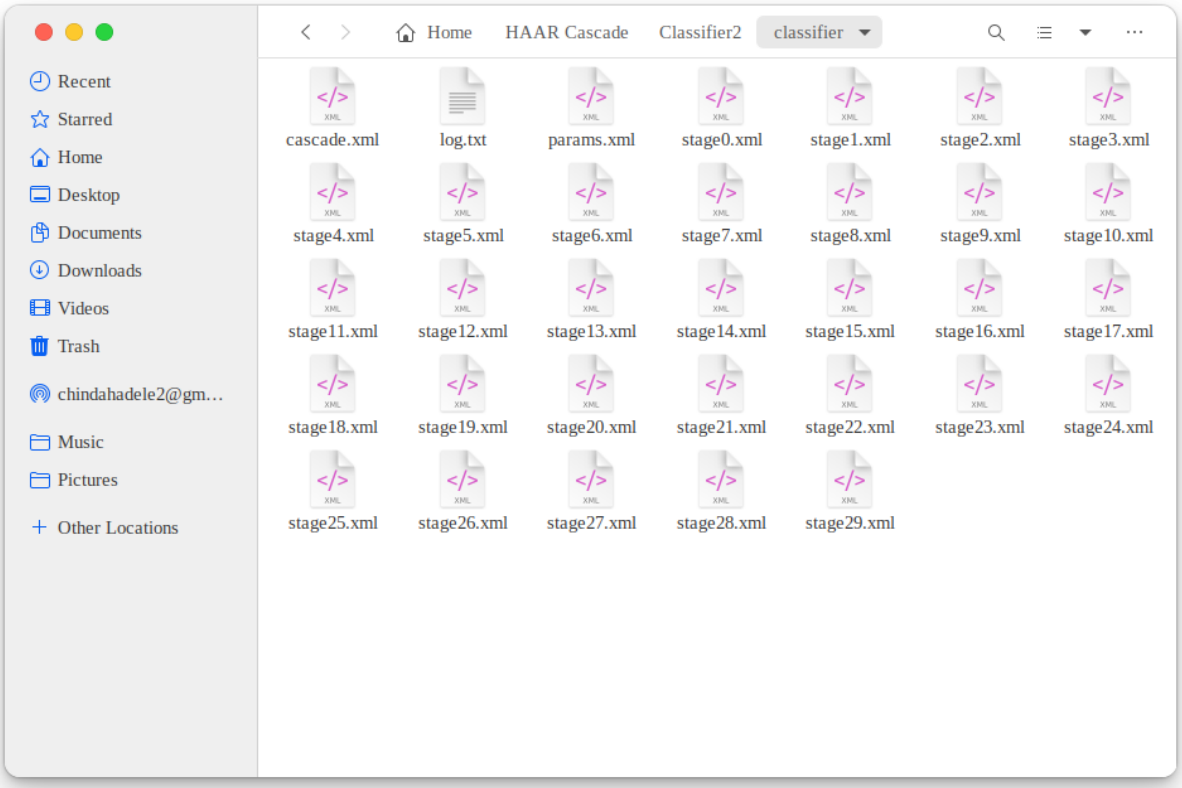

Рисунок 12. Внутренняя папка классификатора

“stage#.xml” файлы - это временные файлы, которые больше не понадобятся. “params.xml” содержит параметры, которые мы использовали для обучения. “cascade.xml” - это фактический каскадный классификатор, и если обучение завершилось успешно, то у нас должен быть этот файл внутри папки классификатора. “neg.lst”, “pos.lst” и “pos_samples.vec” - это временные файлы, созданные для обучения классификатора, и они могут быть удалены без какого-либо эффекта.

V. ТЕСТИРОВАНИЕ КЛАССИФИКАТОРА

Чтобы протестировать наши классификаторы, нам нужно перейти на вкладку "Тест" на панели вкладок вверху и установить параметры, как описано ниже, и, наконец, нажать кнопку "Пуск". Сначала мы выбираем наш каскадный классификатор, используя кнопку Обзора вверху. Мы также можем вручную ввести путь к файлу cascade XML в поле Cascade Classifier XML. Этот каскадный классификатор будет использоваться для обнаружения в изображениях и / или видео. Далее мы можем выбрать одну из следующих настроек ввода, и нам нужно задать путь в соответствии с этой опцией:

1. Одно изображение: Одно изображение будет использоваться в качестве сцены, в которой будет выполняться обнаружение. В этом случае путь должен указывать на один файл изображения. (Можно выбрать только поддерживаемые изображения.)
2. Изображения в папке: Для тестирования будет использоваться папка, содержащая много изображений. В этом случае путь должен быть задан для папки, содержащей одно или несколько изображений сцены.
3. Видео: Для тестирования классификатора будет использоваться видеофайл. В этом случае путь должен указывать на видеофайл, который будет использоваться в качестве входных данных.

После настройки входных параметров настало время для настройки выходных параметров. Тип вывода и путь должны быть установлены в соответствии с тем, что было выбрано в настройках ввода. Смотреть ниже:

1. Изображения в папке: Обнаруженные объекты будут сохранены в нескольких изображениях внутри папки. В этом случае путь должен быть задан для папки. Обратите внимание, что этот параметр можно использовать со всеми возможными параметрами ввода.
2. Видео: Будет создан видеофайл с обнаруженными объектами, выделенными красным прямоугольником. В этом случае путь должен указывать на видеофайл, который будет создан после завершения теста.

Обратите внимание, что эта опция будет работать только в том случае, если в настройках ввода выбрано видео.

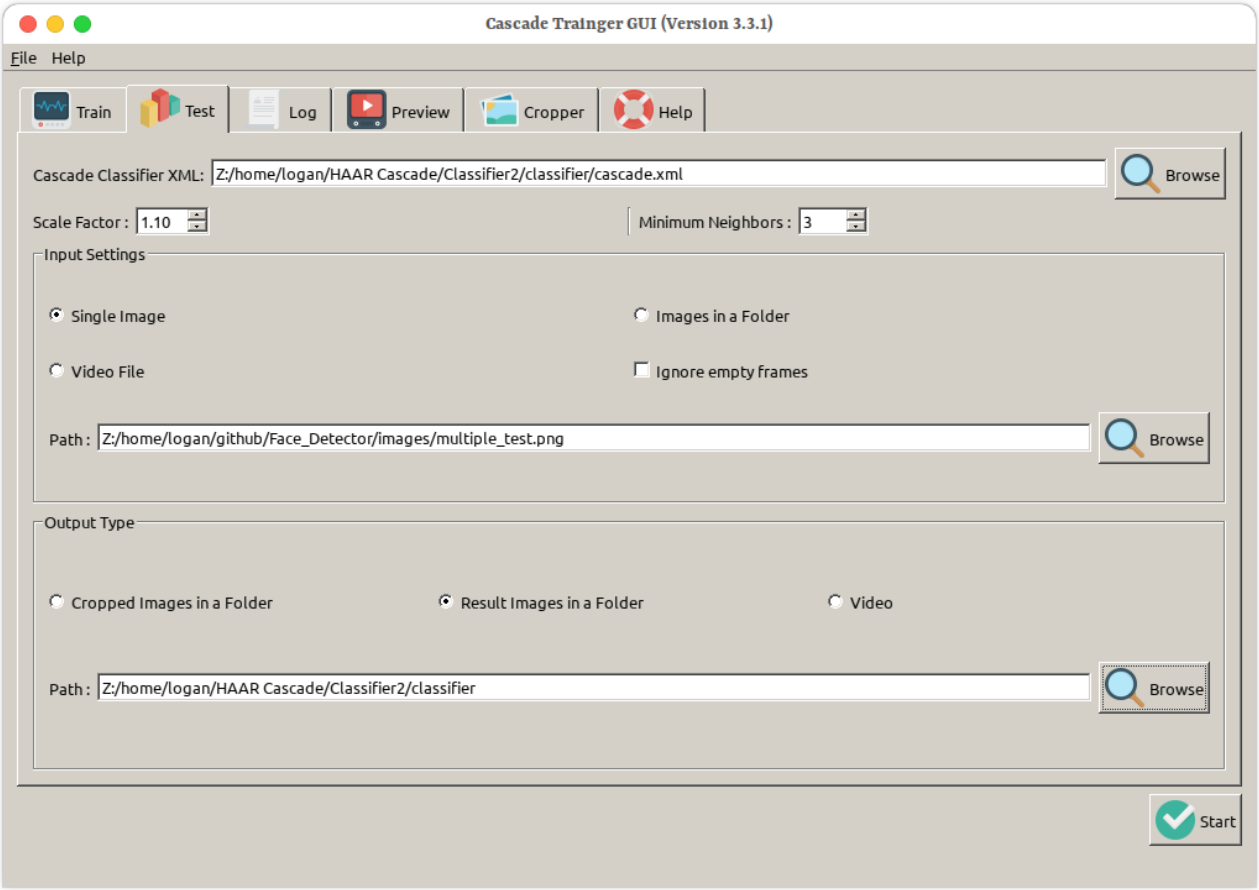

Рисунок 13. Вкладка Test

После нажатия на кнопку "Start" начнется тест. Мы видим индикатор выполнения и следующий экран предварительного просмотра с результатами.

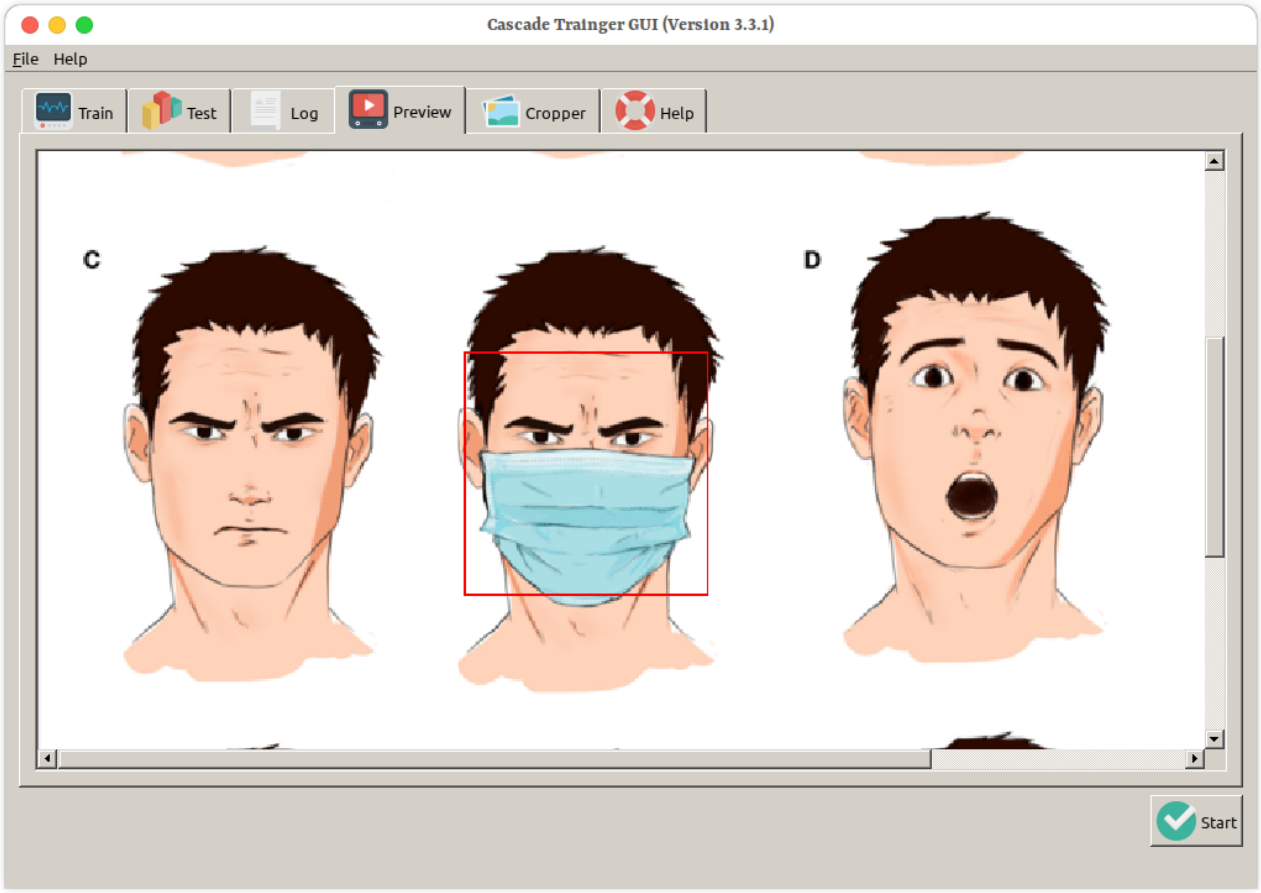

Рисунок 14. Вкладка Preview

После завершения процесса тестирования проверьте выходную папку, чтобы проанализировать точность классификатора. В первый раз обучения мы получаем некоторые “ложноположительные”, а также “ложноотрицательные” ошибки при обнаружении. В этом случае нам нужно снова переобучить наш классификатор, выполнив все описанные выше шаги. Однако во время переподготовки мы должны сделать одну вещь: нам нужно обучить классификатор этим “ложноположительным” и “ложноотрицательным” ошибкам. Это означает, что во время переподготовки нам нужно поместить “ложноположительные” области, которые на самом деле являются отрицательными образцами, в папку “n” в папке классификатора, а “ложноотрицательные” на самом деле являются изображениями лиц, которые нам нужно поместить в папку “p” в папке классификатора. Кроме того, таким образом, мы можем снизить эту частоту ошибок.

```python
#!/usr/bin/env python3
from tkinter.ttk import Frame
import cv2
import numpy as np
from PIL import Image
import os
import  random

trained_faces_data = cv2.CascadeClassifier("cascade.xml")
#* Choose image to detect
img1 = cv2.imread("images/data_test1.jpg")
img2 = cv2.imread("images/data_test2.jpeg")
img3 = cv2.imread("images/multiple_test.png")
#* Convert the data to greyscale
greyscaled_img1 = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
greyscaled_img2 = cv2.cvtColor(img2, cv2.COLOR_BGR2GRAY)
greyscaled_img3 = cv2.cvtColor(img3, cv2.COLOR_BGR2GRAY)
 #* Detect faces
masked_coordinates1 = trained_faces_data.detectMultiScale(greyscaled_img1)
masked_coordinates2 = trained_faces_data.detectMultiScale(greyscaled_img2)
masked_coordinates3 = trained_faces_data.detectMultiScale(greyscaled_img3)
#* Draw rectangles around faces
colors = ((256,0,0), (0,256,0), (0,0,256))

for (x, y, w, h) in masked_coordinates1:
    color = random.choice(colors)
    cv2.rectangle(img1, (x, y),(x+w, y+h), color, 2)

for (x, y, w, h) in masked_coordinates2:
    color = random.choice(colors)
    cv2.rectangle(img2, (x, y),(x+w, y+h), color, 2)

for (x, y, w, h) in masked_coordinates3:
    color = random.choice(colors)
    cv2.rectangle(img3, (x, y),(x+w, y+h), color, 2)

cv2.imshow("Masked face", img1)
cv2.imshow("Unmasked face", img2)
cv2.imshow("Multiple faces", img3)
cv2.waitKey()
cv2.destroyAllWindows()
```

Рисунок 15. Тестирование классификатора с использованием кода Python

Список литературы

[1] Распознавание лиц с помощью HAAR-Каскада - Документация [Электронный ресурс]– URL https://towardsdatascience.com/face-detection-with-haar-cascade-727f68dafd08

[2] Набор данных для обнаружения маски лица - Документация [Электронный ресурс]– URL https://www.kaggle.com/datasets/tapakah68/medical-masks-part1

[3] Каскад Хаара Объяснен - Документация [Электронный ресурс] – URL http://www.willberger.org/cascade-haar-explained/

[4] Каскадный классификатор OpenCV - Документация [Электронный ресурс] – URL https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html